MARKOV MODEL

Gábor Visky
Researcher
NATO Cooperative Cyber Defence
Centre of Excellence

03.29.2022

MARKOV MODEL

Model to describe complex processes

Named after Andrey Markov (1856-1922)

Markov Processes: A memoryless chain of states.

Memoryless: (Markov Assumption) The next state depends only on the the current state.

$$p(x_{t+1} \mid x_0, \ldots, x_t) = p(x_{t+1} \mid x_t)$$

JOINT DISTRIBUTION

Stochastic processes: A processes, in which the **state evolution is random** over time.

Any joint distribution over sequences of states can be factored according to the chain rule into a product of conditional distributions:

$$p(x_0, x_1, \dots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_0, \dots, x_{t-1})$$

EXAMPLE

What is the probability of a sentence: ”The cat sat on the mat ?”

p(The cat sat on the mat) =

p(The) ×

p(cat | The) ×

p(sat | The cat) ×

p(on | The cat sat) ×

p(the | The cat sat on) ×

p(mat | The cat sat on the)

Problem: Infeasible amount of data necessary to learn all the statistics reliably.

MARKOV PROCESS

$$p(x_{t-1}, x_{t+1} \mid x_t) = \cancel{p(x_{t-1} \mid x_t)}\, p(x_{t+1} \mid x_t)$$

Let us suppose that the future is independent of the past given the present. Can we, in the real life?

$$p(x_{t+1} \mid x_0, \ldots, x_t) = p(x_{t+1} \mid x_t)$$

Combining the Markov assumption with the chain rule:

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$

Instead of:

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_0, \ldots, x_{t-1})$$

THE SENTENCE AGAIN

$p(\text{The cat sat on the mat}) =$

$p(\text{The}) \times p(\text{cat} \mid \text{The}) \times p(\text{sat} \mid \text{cat}) \times p(\text{on} \mid \text{sat}) \times p(\text{the} \mid \text{on}) \times p(\text{mat} \mid \text{the})$

MORE
COMPUTATIONAL
FRIENDLY

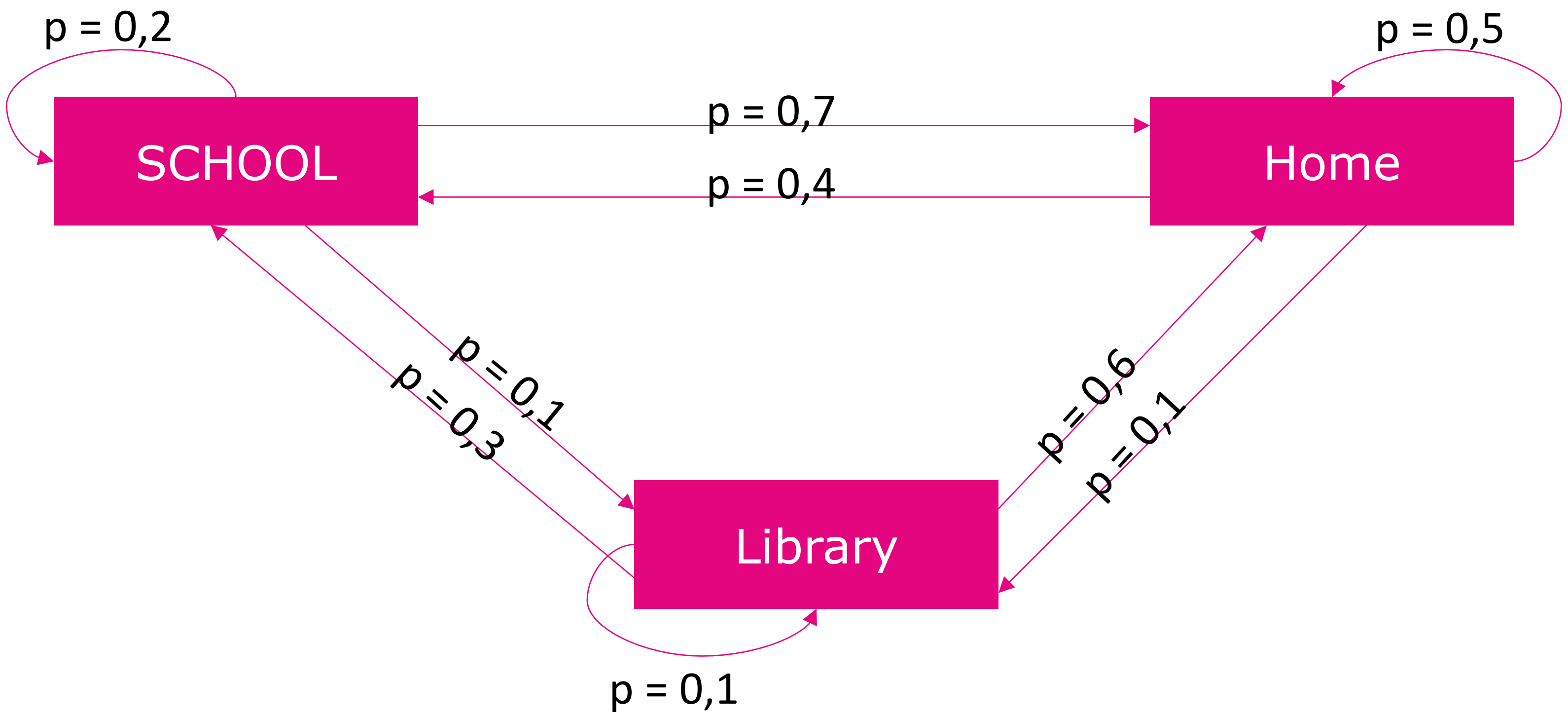
MARKOV CHAIN

The sequence generated by a Markov process is called the Markov chain.

Usually it is assumed that the Markov chain is time-invariant or stationary - this means that the probabilities $p(x_t \mid x_{t-1})$ do not depend on time.

For example in language modelling the probability $p(\text{the} \mid \text{on})$ does not depend on the positions of these words in the sentence.

EXAMPLE

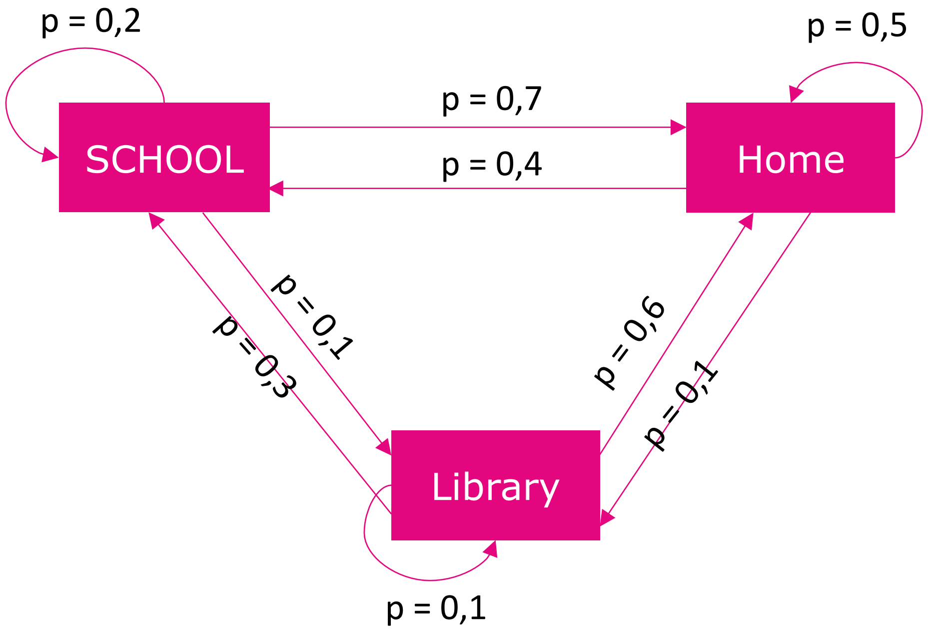


States

Transactions

Transaction probability

TRANSACTION MATRIX



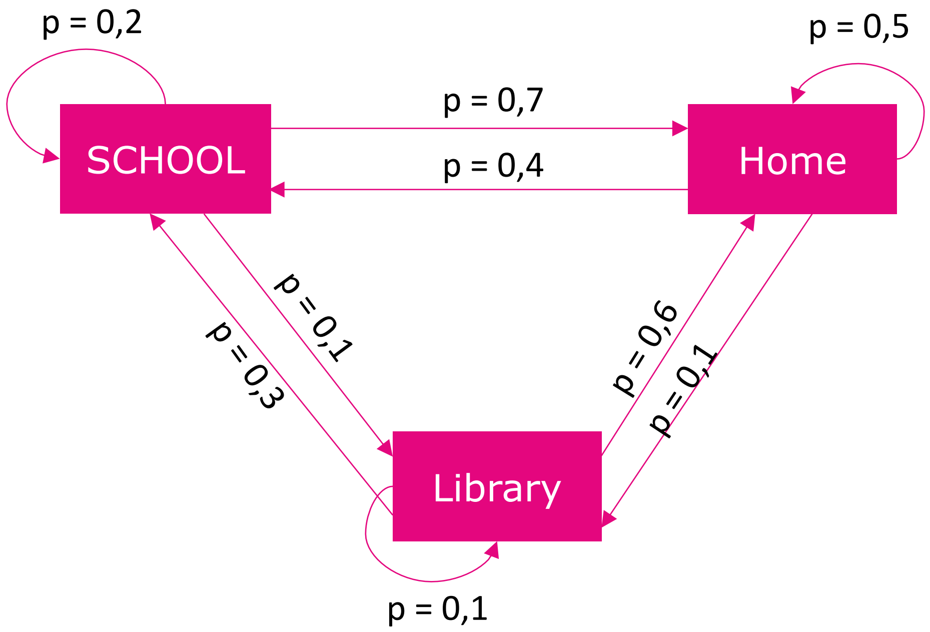
A stationary Markov model with N states can be described by an N × N transition matrix:

$$Q = \begin{bmatrix} q_{11} & \dots & q_{1N} \\ \dots & \dots & \dots \\ q_{N1} & \dots & q_{NN} \end{bmatrix} \qquad q_{ij} \geq 0, \qquad \sum_{i=1}^{N} q_{i,j} = 1, \text{for all } j$$

where $q_{ij} = p(x_t = i \mid x_{t-1} = j)$

$$Q = \begin{bmatrix} 0,2 & 0,1 & 0,7 \\ 0,3 & 0,1 & 0,6 \\ 0,4 & 0,1 & 0,5 \end{bmatrix}$$

STATE DIAGRAM

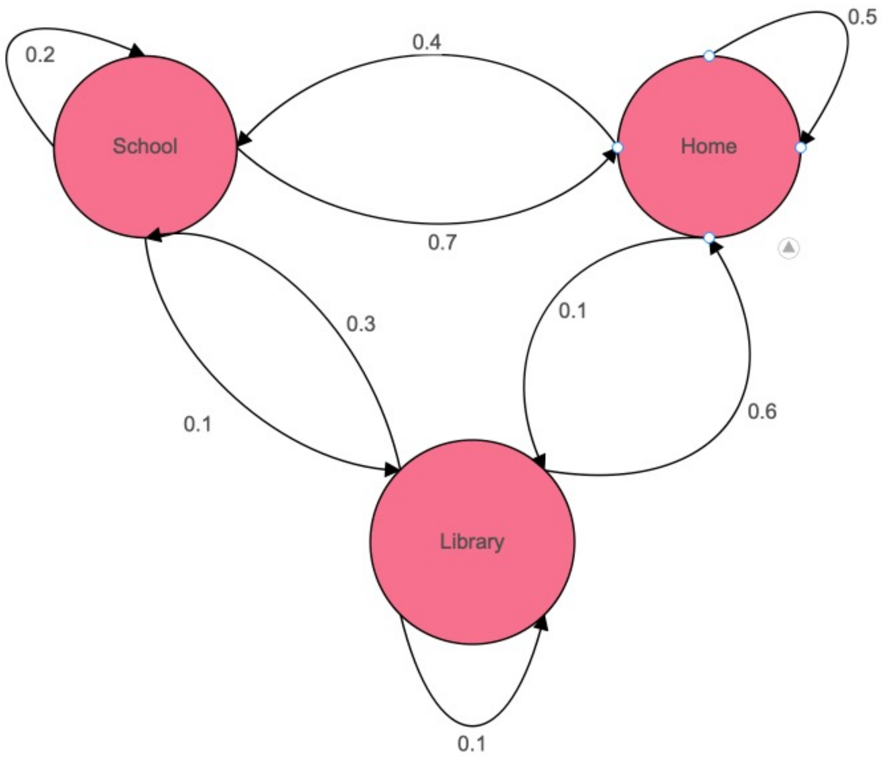


$$\begin{array}{ccc} & \text{COL:FROM} & \\ Q = & \begin{bmatrix} 0,2 & 0,1 & 0,7 \\ 0,3 & 0,1 & 0,6 \\ 0,4 & 0,1 & 0,5 \end{bmatrix} & \text{ROW:TO} \end{array}$$

State transition matrices can be visualized with a state transition diagram.

State transition diagram is a directed graph where arrows represent legal transitions.

Drawing state transition diagrams is most useful when N is small and Q is sparse.



GRAPHICAL MODELS

A way of specifying conditional independencies.

Directed graphical model: DAG.

Nodes are random variables.

A node’s distribution depends on its parents.

Joint distribution: $p(X) = \prod_i p(x_i \mid \mathsf{Parents}_i)$

A node’s value conditional on its parents is independent of other ancestors.

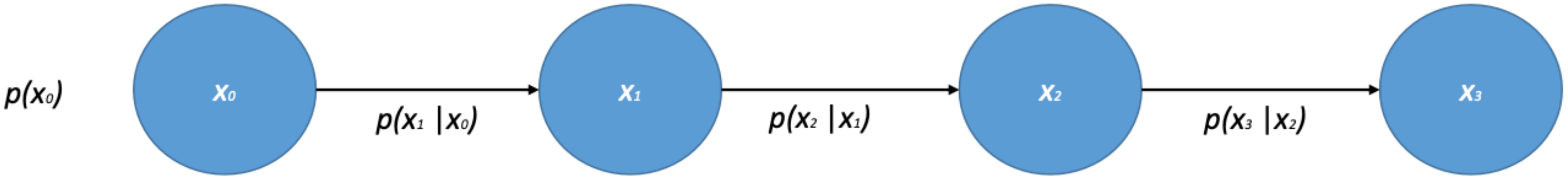
MARKOV CHAIN AS GRAPHICAL MODEL

$$p(x_0, x_1, \ldots, x_T) = p(x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})$$

Graph interpretation differs from state transition diagrams:

Nodes represent state values at particular times

Edges represent Markov properties



MARKOV CHAIN TRAINING

Training data is given in the form of sequences (from observations for example)

Number of occurrence of any two consecutive values can be counted. (How many “The -> cat” pair exists?)

Probability: $p(The|cat) = \frac{p(The\ cat)}{p(The)} = \frac{Count(The\ cat)}{Count(The)}$

In general, if $N_{i,j}$ is the number of times the value i is followed by the value j:

$$p(x_t = j \mid x_{t-1} = i) = \frac{p(x_{t-1} = i, x_t = j)}{p(x_{t-1} = i)} = \frac{N_{i,j}}{\sum_j N_{ij}}$$

MARKOV CHAIN ORDER

First-order Markov model was discussed until now.

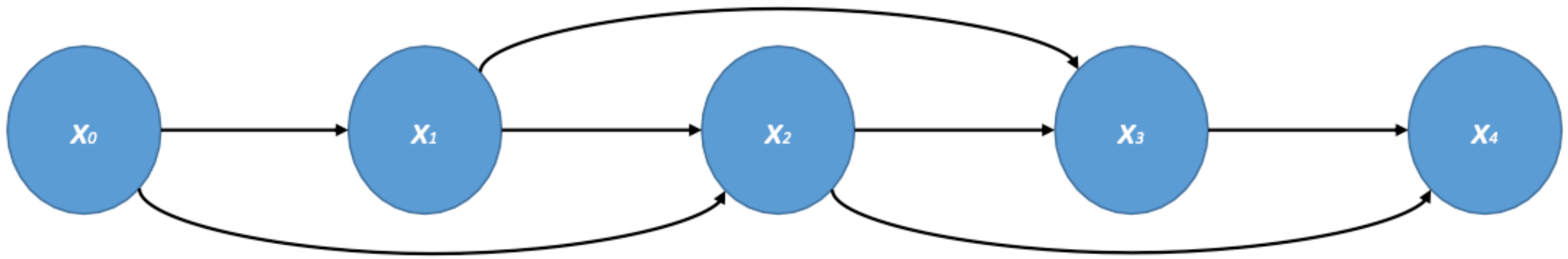
It is also called bigram model (especially in language modelling)

The marginal probabilities $p(x_t)$ called **unigram** probabilities

In the unigram model all the variables are independent: $p(x_0, x_1, \ldots, x_T) = \prod_t p(x_t)$

Higher order Markov chains: a second order model operates with **trigrams**:

$$p(x_t \mid x_0, \ldots, x_{t-1}) = p(x_t \mid x_{t-2}, x_{t-1})$$



PROBLEMS

Few realistic sequential processes directly satisfy the Markov assumption.

Markov chains cannot capture long-range correlations between observations.

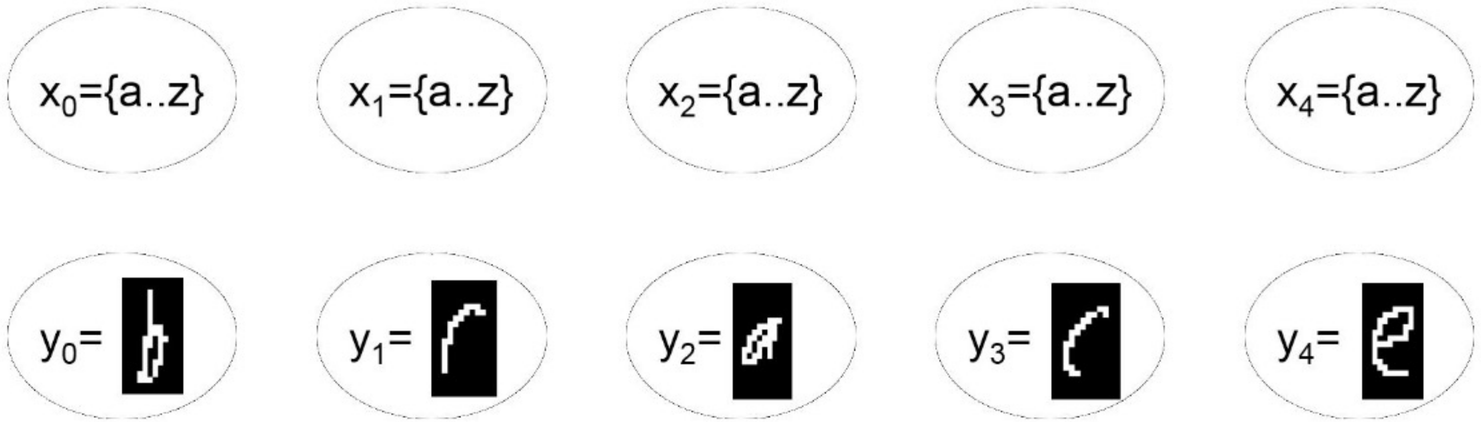
Increasing the order leads the number of parameters to blow up.

The data is the noisy observation of this process.

Solution: the hidden Markov models (HMM).

In HMM there is an **underlying hidden process** that can be **modelled with** a first-order **Markov chain**.

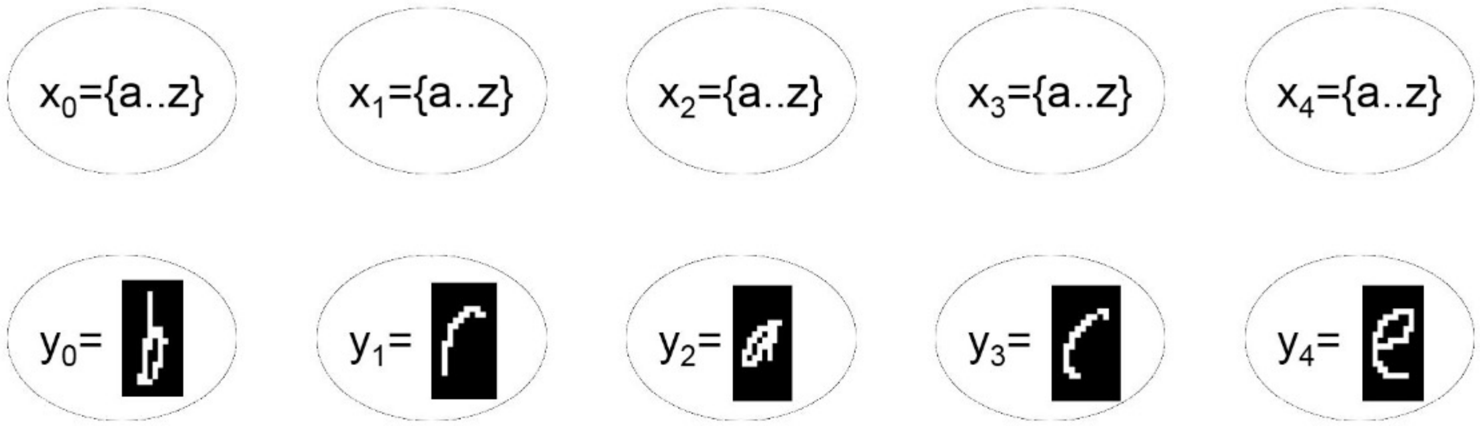
EXAMPLE – HANDWRITTEN CHARACTERS



What is the hidden process?

What can be modelled with first order Makarov chain?

HMM SPECIFICATION



There are three distributions:

$$p(x_0)$$

$$p(x_t \mid x_{t-1}), \quad t = 1, \ldots, T$$

$$p(y_t \mid x_t), \quad t = 1, \ldots, T$$

JOINT DISTRIBUTION

$$p(x_0, \ldots, x_T) \mid y_0, \ldots, y_T) \propto p(x_0)p(y_0 \mid x_0) \prod_{t=1}^{T} p(x_t \mid x_{t-1})p(y_t \mid x_t)$$

DETAILS

Compute marginal probabilities of hidden variables.

Filtering (on-line):compute the belief states $p(x_t \mid y_0, \ldots, y_t)$

Smoothing: (off-line, using all the evidences) compute the probabilities: $(x_t \mid y_0, \ldots, y_T)$

Find the most likely sequence of hidden variables - Viterbi decoding. (weather/mood example)

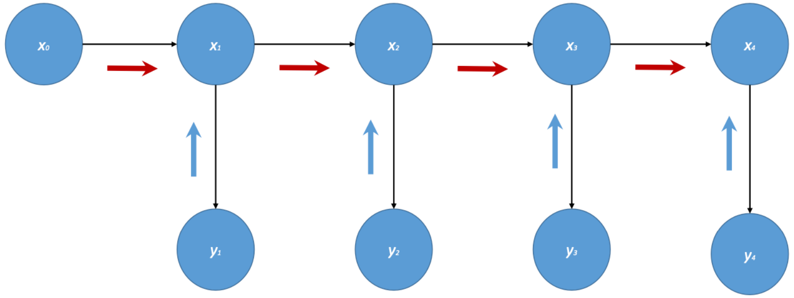
FILTERING

Computing $p(x_t \mid y_0, \ldots, y_t)$ is called filtering, because it reduces noise in comparison to computing just $p(x_t \mid y_t)$.

Filtering is done using forward algorithm.

Forward algorithm uses dynamic programming - this means the algorithm is recursive but we reuse the already done computations.

Forward algorithm

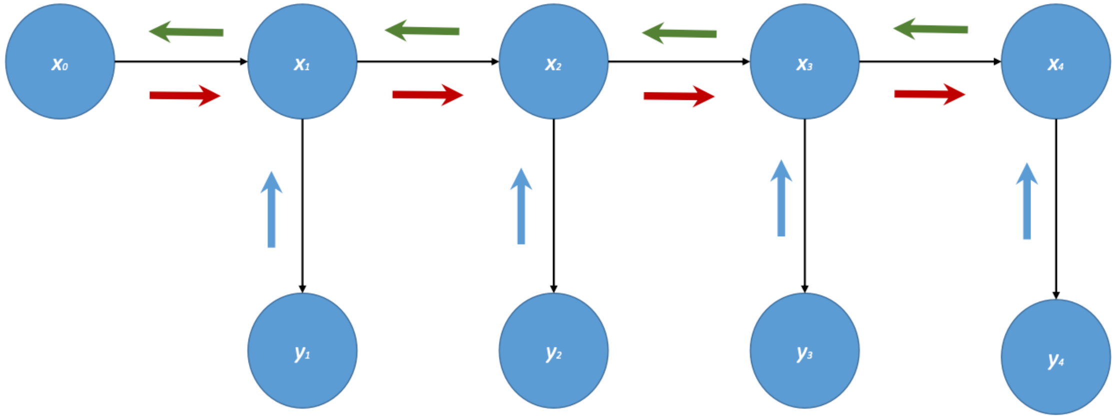


Input:
- Transition matrix
- Initial state distribution
- Observation matrix containing probabilities $p(y_t \mid x_t)$

Compute the forward probabilities:

$$\alpha_t(x_t) = p(x_t \mid y_{1:t}) = \frac{1}{Z_t} p(y_t \mid x_t) \sum_{x_{t-1}} p(x_t \mid x_{t-1} \alpha_{t-1}(x_{t-1}))$$

SMOOTHING



Smoothing computes the marginal probabilities $p(x_t \mid y_{1:T})$ off-line, using all the evidence

It is called smoothing, because conditioning on the past and future data, the uncertainty will be significantly reduced.

Smoothing is performed using forward-backward algorithm.

FORWARD-BACKWARD ALGORITHM

Break the chain into past and future:

$$p(x_t = j \mid y_{1:T}) \propto p(x_t = j, y_{t+1:T} \mid y_{1:t})$$
$$\propto p(x_t = j \mid y_{1:t}) p(y_{t+1:T} \mid x_t = j)$$

Compute the forward probabilities in traditional way: $\alpha_t(x_t) = p(x_t = j \mid y_{1:t})$

Compute the backward probabilities: $\beta_t(x_t) = \frac{1}{Z_t} \sum_{x_t} p(x_{t+1} \mid x_t) p(y_{t+1} \mid x_{t+1}) \beta_{t+1}(x_{t+1})$

OPTIMAL **STATE** ESTIMATION

Compute the smoothed posterior marginal probabilities: $p(x_t \mid y_{1:T}) \propto \alpha_t(x_t)\beta_t(x_t)$

Probabilities measure the posterior confidence in the true hidden states.

Takes into account both the past and the future.

OPTIMAL **SEQUENCE** ESTIMATION

Viterbi algorithm computes: $\hat{x} = \arg\max p(x_0, \ldots, x_t \mid y_1, \ldots y_T)$

Using dynamic programming it finds recursively the probability of the most likely state sequence ending with each x_t:

$$\gamma_t(x_t) = \max_{x_1,\ldots,x_{t-1}} p(x_1, \ldots, , x_t \mid y_{1:t})$$

$$\propto p(y_t \mid x_t) \left[\max_{x_{t-1}} \quad p(x_t \mid x_{t-1}) \gamma_{t-1} x_{t-1} \right]$$

A backtracking procedure picks then the most likely sequence.

LEARNING HMM

If latent state sequence is available during training, then the transition matrix, observation matrix and initial state distribution can be estimated by normalized counts.

$$\hat{q}_{i,j} = \frac{n(i,j)}{\sum_k n(k,j)}$$

$$\tau_i = \{t \mid x_t = i\}$$

$$\hat{\theta}_i = \frac{1}{\mid \tau_i \mid} \sum_{t \in \tau_i} y_t$$

Typically the hidden state sequences are not known.

EM algorithm is used, that iteratively maximizes the lower bound on the true data likelihood.

E-step: Use current parameters to estimate the state using forward-backward.

M-step: Update the parameters using weighted averages.

DO YOU HAVE ANY QUESTIONS?